VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Implementace tradičních algoritmů pro porovnávání obrazu
Image Comparing Traditional Algorithm Implementation

2010 Tomáš Oguršák

Prohlašuji, že jsem tuto práci vypracoval sám a uvedl jsem veškeré literární zdroje, ze kterých jsem čerpal.

Datum:
Podpis:

Poděkování

Hlavní poděkování patří vedoucímu práce, který mi ochotně pomáhal při tvorbě práce a poskytoval mi veškeré potřebné materiály k vypracování.

Abstrakt

Tato Bakalářská práce se zabývá aplikací pro editování fotografií. První část obsahuje jednoduchý popis již existujících aplikací na našem trhu. Druhá část se zabývá analýzou a implementací systému pro porovnávání fotografií a jednoduchou editaci fotografií s rozšířením o vykonávání sekvencí. Analýza je zaměřena hlavně na detaily případu užití a sekvenční diagramy. Z implementace jsou vyzdviženy nejdůležitější části kódu pro hlavní funkce.

Klíčová slova

bakalářská práce, sekvence, skript, komponenta, systém, aplikace, MainMenu, switch, TreeView, TextBox, CheckBox

Abstract

This baccalaureate work deal with application for editing photographs. Forepart is simple description already going application on our market. The second part deals with analysis and implementation of system for comparing photographs and simple editing photographs with extension of performing of sequencies. Analysis is largely bent on detail of case use and sequential diagrams. From implementation is upheaval the most important part of code for main function.

Keywords

Bachelor thesis, sequence, script, component, systém, application, MainMenu, switch, Treeview, TextBox, CheckBox

Seznam použitých symbolů a zkratek

XML - Extensible Markup Language/rozšiřitelný značkovací jazyk
HDD - hard disk drive / pevný disk
MP3 - formát ztrátové komprese zvukových souborů, založený na kompresním algoritmu MPEG
CS4 - Creative Suite 4

Obsah

1. Úvod

Cílem této bakalářské práce je zmapovat již existující trh s nástroji pro správu fotografií. Vybrat ty nejvhodnější z nich a vytvořit porovnání s popisem funkcí. Následně zhodnotit dle vlastních poznatků jejich klady a zápory jak pro funkce, které nástroje obsahují tak pro přijatelnost prostředí a složitosti pro běžné uživatele.

V druhém kroku dle nastudovaného materiálu vytvořit vlastní aplikaci pro správu fotografií, která by měla obsahovat co nejvíce základních operací nad fotografiemi s přijatelným uživatelským prostředím na ovládání. Dále k této aplikaci vytvořit kompletní analýzu a dokumentaci, aby mohla být práce předána navazujícímu řešiteli k rozšiřování.

2. Současný stav aplikací na trhu

V dnešní době existuje spousta profesionálních či amatérských aplikací pro úpravu fotografií. V následující kapitole zkusíme lehce naznačit a popsat ty nejvýznamnější z nich. Zaměříme se na tři hlavní kategorie, a to prohlížeče fotografií, editační programy a programy pro hromadné úpravy. Těchto aplikací je bezpochyby na trhu nejvíce.
Jejich rozdíly ve funkčnostech se postupem doby postupně smazávají, a proto se prodejci snaží svého zákazníka získat různými způsoby, jako jsou:

- jednoduchostí ovládání
- přívětivým vzhledem
- rychlostí běhu či možnostmi přidávání různých plug-in filtrů

Poslední dobou se začala také implementovat funkce pro hromadné úpravy. Bez této funkce se málokterá aplikace ujme na trhu. Nyní si rozebereme několik aplikací podrobněji.

2.1 IrfanView

Tento program je nejspíše jeden z nejoblíbenějších prohlížečů vůbec a jeho oblíbenost u běžných uživatelů neustále narůstá. Už při instalaci může překvapit uživatele množstvím typů souborů, které podporuje. Pokud však zde nenajdete požadovaný typ, který byste vyžadovali, je možnost jej zařadit díky zásuvným modelům, kterých je pro tuto aplikaci velká spousta.
Program má také jednoduché prostředí pro editaci obrázků. Toto prostředí nabízí velmi malé množství funkcí, které si dnešní uživatel přeje, ale zase je zde možnost díky pluginům si zde cokoliv doinstalovat.

Jako velký plus a originalitu bych zde viděl vytvoření vlastní slideshow. Stačí pouze poskládat fotografie za sebe, na pozadí spustit jakoukoliv písničku v MP3 formátu a nastavit pár dalších vlastností. Výhodou je, že výsledná slideshow je spustitelný program a proto je možné jej spouštět i na počítačích, které nemají tento program nainstalován.

Samozřejmě zde také najdeme hromadnou úpravu nad fotografiemi. Můžeme například změnit veškeré své fotografie z nekomprimovaného tvaru do komprimovaného, či pouze provádět jednoduché úpravy jako přejmenovávání, ořezy, kontrasty atd.
Další doplňkové funkce jsou automatické skenování, kdy program dokáže komunikovat se skenerem a můžete tak přímo z něj pořizovat obrázky. Další doplňkové funkce jsou přehrávání videí či hudby. Tyto funkce jsou ale velice jednoduché a občas se stane, že mají problém a program havaruje. Poslední doplňkovou funkcí je pořizování snímků obrazovky.

2.2 ACDSee

ACDSee je sharewarový software pro správu a editaci fotografií a obrázků vyvíjený ACD Systems pro Windows. Je k dispozici v několika různých verzích pro domácí a profesionální použití. Skládá se ze dvou základních produktů; Photo Manager a Photo Editor.

Photo Manager

Stávající verze je 10.0 (build 239). Kromě obvyklých náhledů složek a konverzí souborů patří ke klíčovým prvkům ACDSee prezentace snímků (slide show), CD / DVD vypalování, tvorba HTML

galerie, indexování metadat snímků jako je EXIF. Jsou možné i drobné úpravy obrázků jako ořezávání, škálování a otáčení. Program je k dispozici v angličtině, francouzštině, němčině a nizozemštině. Dále je k dispozici dražší verze Pro s více funkcemi.

Photo Editor

Zjednodušený, nedávno zavedený produkt umožňující jednoduché úpravy jako je psaní textu do obrázku, změna velikosti a vytváření dalších vizuálních efektů.

Kritika

Doteď ACDSee nepodporuje uni-code, pouze multi-byte, takže jména souborů obsahující například asijské znaky se nemusí zobrazit správně. Dále byl ACDSee kritizován za to, že oddíl 2.7 jeho EULA obsahuje zákaz použití ACDSee pro zobrazení pornografického materiálu. [3]

2.3 Photoshop

Jedná se o bezpochyby jeden z nejlepších editorů pro fotografie vytvořený firmou Adobe Systems. První verze tohoto programu vyšla v roce 1990 a dnes máme již 12.tý díl CS5 a to ve dvou provedeních jeden pod označením Adobe Photoshop CS5 a druhý Adobe Photoshop CS5 Extended.

Photoshop family[4]

Jako Photoshop family se označuje „rodina“ následujících sedmi produktů Adobe Systems:

- Photoshop CS4
- Photoshop CS4 Extended
- Photoshop Elements 6.0 for Macintosh
- Photoshop Elements 6.0 for Windows
- Photoshop Elements 6.0 & Adobe Premiere Elements 4.0
- Photoshop Express beta
- Photoshop Lightroom 2

Nevýhody:

- Zásadní nevýhodou Photoshopu je cena, obzvlášť pro Evropu je stanovena hodně vysoko. V internetových obchodech v USA se dá ale Photoshop sehnat až několikanásobně levněji, když se obejdete bez pěkné krabice.
- Na základní úpravy fotek (vyvážení bílé, křivky, ořez, zmenšení, doostření), stačí téměř jakýkoliv foto software a Adobe Photoshop opravdu není třeba vzhledem k jeho vysoké ceně,

Výhody:

- Velký sortiment funkcí pro editaci fotografií. Mezi nejnovější z nich u CS5 patří nový HDR generátor, Přesnější a snadnější výběry, vylepšený bodový retušovací štětec, Camera Raw 6, Optické korekce specifických objektivů, Zrychlení a zjednodušení práce -program je optimalizován pro 64bitové zpracování
- Obrovská uživatelská základna - Ta tvoří tutoriály a obsah diskuzních fór. Kde jinde, než na webech zaměřených na Photoshop, najdete tutoriály a rady profesionálů, od kterých se toho dá nejvíc naučit.
- Nabídka třetích stran - Největší množství placených i zdarma dostupných doplňků je samozřejmě pro Photoshop. Úplně stejné je to s knihami a výukovými kurzy.

- Uplatnění v komerční sféře - Nabité znalosti se dají dobře uplatnit i při hledání zaměstnání, stačí se podívat na nabídky práce, když jde o zpracování obrazových dat, je v drtivé většině případů požadována znalost Photoshopu. Ještě jsem neviděl firmu, která by používala něco jiného.

..

2.3.1 Adobe Lightroom

Adobe Lightroom je samostatná aplikace na import a zpracování RAW fotografií. Podobně jako Photoshop je editor Lightroom navržen na profesionální nasazení, čemuž odpovídají i jeho celkové možnosti

Adobre Lightroom je již delší dobu ohlášený a neméně dlouho dobou vyvíjený projekt uživatelsky přívětivého, ale přitom plně profesionálního editoru na prvotní zpracování RAW fotografií. Vzhled grafického rozhraní programu pochopitelně není zase tolik důležitý, uživatele Windows ale přesto překvapí. Lightroom navíc částečně konkuruje Adobe Photoshopu, jeho zaměření je však poměrně úzce specializované prakticky výhradně do oblasti RAW fotografií a jejich různých barvových a obrazových korekcí. V současné době je k dispozici třetí veřejná betaverze Lightroom pro Mac OS X 10.4.3 a novější, Lightroom beta 3 pro Windows XP (jiné verze Windows nejsou podporovány) je ale novinkou. Ke stažení je verze bez ukázek či s fotografickými ukázkami (má něco přes 100 MB).

3. Požadavky

Tvorba požadavků probíhala dvěma způsoby. První, který byl z hlediska dopadů na systém mnohem podstatnější. Vznikal na konzultacích s vedoucím práce. Zde se domlouvala největší část finálního chování aplikace. Druhý, méně častý způsob, byl využití vlastních poznatků a zkušeností s používáním a nastudováním již existujících aplikací.

3.1 Funkční požadavky:

Funkční požadavky se tvořily během celé práce. Při první konzultaci s vedoucím jsme si hrubě připravili hlavní požadavky, které by měla aplikace obsahovat. Ovšem většinu z nich jsme mnohokrát upravovali vhledem k postupnému přidávání funkcí a jejich vhodnému zapracování.

3.1.1 Požadavky pro aplikaci

Jedná se o jednotlivé požadavky, které bude obsahovat aplikace. Především o souhrn možností, které bude finální aplikace nabízet.

- systém umožní adresářové procházení
- systém umožní náhledy fotografií ve vybraném adresáři
- systém umožní otevírání a ukládání fotografií
- systém umožní základní úpravy fotografií jako změna velikosti, filtry, deformace, jas atd.
- systém umožní postupné zaznamenávání kroků úprav a jejich následnou modifikaci
- systém umožní ukládání, editaci a spouštění sekvencí kroků
- systém umožní hromadnou aplikaci sekvencí na vybraný adresář fotek
- systém umožní hromadné přejmenovávání fotografií ve vybraném adresáři
- systém umožní porovnávání fotografií

3.2 Nefunkční požadavky:

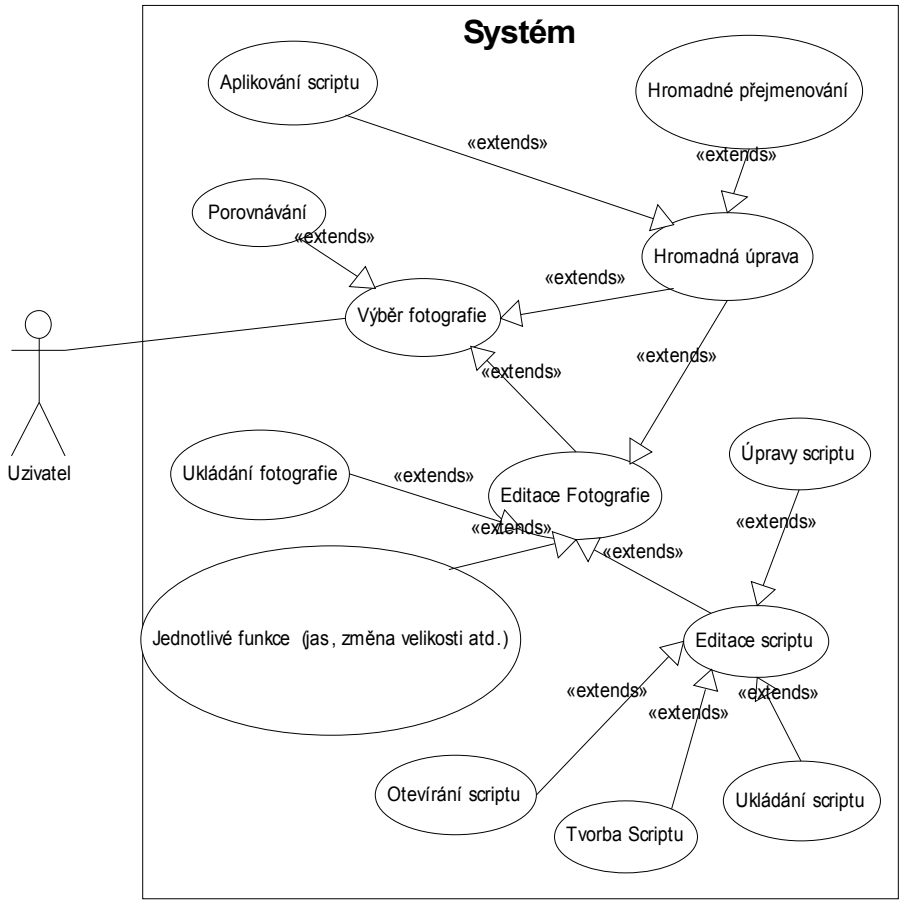
Jedná se o požadavky, pomocí kterých bude systém fungovat, v čem bude systém naprogramován a co bude ke svému chodu využívat. Jediné o čem jsme se rozhodovali bylo, v jakém jazyce bude systém naimplementován.

1. Systém bude naprogramován v C# .NET
2. Systém bude ukládat sekvence funkcí ve formátu XML.

4. Diagramy případu užití

Funkční specifikace je v jazyce UML řešena prostřednictvím diagramu případu užití (Use Case Diagram). Tyto diagramy definují vztahy mezi případy (někdy se také hovoří o scenáriích) užití systému a aktéry stojící vně systému. Případy užití specifikující vzory chování realizované softwarovým systémem. Každý případ užití lze chápat jako posloupnost vzájemně navazujících transakcí vykonaných v dialogu mezi aktérem a vlastním softwarovým systémem. Jako aktéry pak definujeme uživatele či jiné systémy, kteří budou vstupovat do interakce s vyvíjeným softwarovým systémem. [1] Všechny diagramy případu užití naleznete v příloze číslo 2.

4.1 Diagram Případu užití pro aplikaci



Obrázek č. 1. :Diagram případu užití pro aplikaci

Z případu užití na obrázku č.1 lze rozpoznat jaké kroky bude moct uživatel aplikace provádět. Po spuštění samotného softwaru se uživateli zobrazí hlavní strana, kde po levé straně bude mít stromovou strukturu disku. Po výběru adresáře se v pravé straně aplikace zobrazí thumbnaily (náhledy) fotografií, které tato složka obsahuje. Pochopitelně pokud složka neobsahuje žádné fotografie, tak se nezobrazí žádný náhled.

Po kliknutí na náhled se vybraný obrázek přepne do své plné velikosti. Zde můžeme provádět jednoduché operace, které nabízí většina standardních volně dostupných prohlížečů obrázků. Tyto aplikace jsou například zmenšení/zvětšení o 10%, otočení o 90 stupňů doprava nebo doleva.
Další nejdůležitější částí je přepnutí do samotného editačního režimu. Tento nám nabízí velké množství funkcí, které lze provádět. Jedná se o již zmiňované funkce při prohlížení obrázku, ovšem s rozšířenými vlastnostmi. Například otáčet lze o vybrané stupně a převracet je. U velikosti si můžeme zvolit, jakým způsobem chceme zvětšovat či zmenšovat, můžeme nastavovat samotnou velikost obrázku v pixelech či procentech atd. Jsou zde také na-implementovány funkce jasu, kontrastu, rozmazání a šumu či filtrace obrázku. Tyto všechny funkce si popíšeme později. Během celého programu lze také spouštět hromadou úpravu či porovnávání fotografií. Tuto si také popíšeme v dalších kapitolách.

4.2 Detaily případu užití

Tyto detaily rozšiřují předešlé diagramy případu užití a to tím způsobem, že přesně rozepíšou, jak se v praxi v systému bude chovat jednotlivý případ užití. Všechny detaily případu užití naleznete v příloze na CD.

4.2.1 Detaily případu užití pro hromadnou úpravu

Případ užití: Hromadná úprava
ID:S1
Účastníci: Uživatel
Vstupní podmínky: Je zobrazena hlavní strana, nebo editační strana
Tok událostí: 1. Systém zobrazí editační prvky pro zadávání aktuálního adresáře. 2. Systém doplní cestu cílového adresáře přidáním adresáře „nový“ do aktuálního adresáře, ale umožní jeho změnu 3. <<include>> Systém umožní změnu fotografií • Systém zobrazí editační pole pro výběr názvů fotografií s nápovědou • Systém kontroluje, zda nevznikne kolize v názvech souborů 4. <<include>> Systém umožní zvolení skriptu, který bude aplikován 5. Uživatel spustí provádění 6. Systém postupně načítá fotografie, aplikuje na ně skript, upraví název a uloží do cílového adresáře
Následné podmínky: Fotografie z aktuálního adresáře jsou upraveny a uloženy do cílového adresáře

Tabulka č. 1.: Detail případu užití – hromadná úprava

V tomto případu užití, kde se jedná o spouštění hromadné úpravy, který lze vidět v tabulce č.1 jde vidět, jak postupně bude systém pracovat, aby nám vznikl požadovaný cíl. Hromadná úprava lze spouštět v jakékoli fázi samotné aplikace. Spouští se v MainMenu pod položkami soubor → hromadná úprava. Poté se nám zobrazí okno ve kterém vybereme složku, nad kterou chceme úpravu provádět a vybere, zda chceme procházet i její podsložky či nikoliv. Následně se nám automaticky do editačního panelu pro cílový adresář doplní stejná adresářová cesta jako cesta vybraná pro aplikování skriptu, která bude doplněna o koncovku „/novy“. Cílovou cestu si také můžeme upravit podle potřeb.
Dále můžeme zvolit, zda chceme nějakým způsobem měnit názvy fotografií. K tvorbě je přiřazena jednoduchá nápověda, která ukazuje, jaké znaky jsou k čemu využívány. Jsou to znaky – popis : @ - původní název souboru, + - inkrementace čísel, & - aktuální datum, % - aktuální čas. Veškeré znaky lze také použít jako text, pokud před ně vložíme lomeno - / .

Pokud chceme použít lomítko jako text a za nim následuje některá z funkcí schovaná pod symbolem, tak se znegování funkce zruší tím, že se zneguje předchozí lomítko lomítkem. Pod odstavcem si uvedeme několik příkladů.

Po vytvoření názvů si zvolíme, zda chceme aplikovat některé z dříve vytvořených posloupností funkcí. Pokud toto zvolíme, tak se nám panel zvětší o okno s náhledem na posloupnost operací vybraného skriptu. Vybereme skript, který chceme aplikovat. Výběr je směřován do adresáře s projektem, kde je směrováno i ukládání skriptů. Po vybrání se nám pro kontrolu zobrazí v poli pro náhled celá postupná posloupnost provádění skriptu v jednotlivých krocích. Po odsouhlasení všech zvolených parametrů začne program postupně vybírat všechny fotografie ze zvoleného adresáře, popřípadě jeho podadresářích, otevírá si je postupně, vloží je do editačního módu, aplikuje vybraný skript, a ukládá pod vybraným změněným názvem. Tohle provádí, dokud má nějaké fotografie.

4.2.2 Detaily případu užití pro výběr fotografie

Případ užití: Výběr fotografie
ID:S2
Účastníci: Uživatel
Vstupní podmínky: Je zobrazena hlavní strana
Tok událostí: 1. systém zobrazí adresářovou strukturu HDD ve stromové struktuře 2. uživatel si zvolí adresář 3. systém zobrazí náhledy všech fotografií ve zvoleném adresáři 4. uživatel může vybrat fotografii kliknutím na náhled 5. uživatel může vybrat fotografii přes menu soubor → otevřít 6. fotografie se zobrazí v normálním náhledu 7. systém zobrazí panel pro jednoduché úpravy fotografie

Následné podmínky:
Fotografie je zobrazena

Tabulka č. 2.: Detail případu užití – výběr fotografie

V tomto detailu případu užití z tabulky č.2 lze vidět, jak aplikace postupuje při vybírání jednotlivé fotografie, na které chceme provádět úpravy či vytvářet samotný skript. Na začátku máme zobrazenou hlavní stranu, u které vidíme v levé části komponentu pro zobrazení a procházení adresářové struktury HDD. V této komponentě jdou také provádět jednoduché funkce jako vytvoření složky, kroky zpět a vpřed, přesunutí do adresáře, kde se nachází projekt, či zobrazení hlavní struktury HDD. Po výběru jednotlivé složky se nám v hlavní části okna zobrazí náhledy jednotlivých fotografií, které jsou obsaženy ve vybraném adresáři.

Pokud jsme si již vybrali fotografii, jednoduše jí vybereme kliknutím na její náhled a fotografie se nám zobrazí v jednoduchém editačním režimu. Fotografii lze také vybrat jednoduše, pokud ji již máme předem vybranou a víme její zdrojovou cestu. V tomto případě stačí pouze v MainMenu vybrat položky soubor → otevřít. Vyskočí nám již známé okno pro výběr souboru, který vyžadujeme. Po výběru a odsouhlasení se nám fotografie opět přepne do jednoduchého režimu. V tomto režimu můžeme provádět jednoduché úpravy jako například: otáčení o 90 stupňů či zvětšování a zmenšování o 10%. Také se můžeme přepnout zpátky na hlavní stranu, otevírání, nebo ukládání obrázku. Dále zde máme už samotné filtry, či přepnutí do editačního režimu. Po výběru z jednoho z nich se nám fotografie přepne do editačního režimu.

4.2.3 Detaily případu užití pro editaci fotografie

Případ užití: Editace fotografie
ID:S3
Účastníci: Uživatel
Vstupní podmínky: Je vybrána fotka a přepnuta do editačního režimu
Tok událostí: 1. systém zobrazí panel s editačními funkcemi 2. uživatel si zvolí funkci 3. systém zobrazí rozšířené vlastnosti vybrané funkce 4. uživatel vybere detaily, funkce, které vyžaduje a potvrdí je 5. sytém zobrazí jak by fotografie vypadala po změně 6. uživatel vybere zda chce změnu uložit či jí stornovat 7. systém umožní zpětné resetování do původního stavu kdykoliv během editace
Následné podmínky: Fotografie je editována

Tabulka č. 3.: Detail případu užití – editace fotografie

V tomto detailu případu užití z tabulky č. 3 můžeme pochopit, jak systém pracuje se samotným editačním režimem. Jako vstupní podmínku zde máme, že jsme si již předem vybrali fotografii, na které chceme aplikovat změny a tu jsme převedli do editačního režimu.

V tomto režimu máme na výběr z šesti hlavních kategorií. Jsou to otáčení, velikost, šum, rozmazání, jas a filtry. Po vybrání jedné z nich se nám zobrazí její rozšířené možnosti. Například u otáčení to je, zda chceme rotovat o předem daný úhel, či chceme otáčet o námi zadané stupně, dále zda chceme aplikovat některé ze zadaných převrácení fotografie. Nebo u velikosti, zda chceme měnit velikost procentuálně, nebo zda si sami zadáme pixely výsledné velikosti a zda podle těchto pixelů chceme obrázek deformovat, nebo vypočítat vhodnou nejbližší velikost a obrázek podle toho změnit. Tyto veškeré změny musíme potvrdit a poté se nám ukáže výsledek na samotné fotografii. Pokud změnu neuložíme, tak i přepnutí mezi funkcemi nám obrázek resetuje do poslední uložené konfigurace. Pokud chceme obrázek po změně uložit, musíme toto potvrdit po každé změně. V neposlední řadě zde máme tlačítko, které nám obrázek resetuje do původního načteného vzhledu, i když jsme jej již nějakým způsobem změnili.

4.2.4 Detaily případu užití pro editaci skriptu

Případ užití: Editace skriptu
ID:S4
Účastníci: Uživatel
Vstupní podmínky: Je vybrána fotka a přepnuta do editačního režimu
Tok událostí: 1. systém zobrazí menu pro vytváření skriptu 2. systém umožní zapnutí či nahrávání skriptu 3. uživatel zvolí nahrávání skriptu 4. systém se dotáže na název skriptu 5. <<include>> Systém umožní ukládání skriptu 6. <<include>> Systém umožní editaci skriptu 7. <<include>> Systém umožní úpravy skriptu 8. <<include>> Systém umožní otevření skriptu 9. systém zobrazí tlačítka na provedení skriptu, smazání části skriptu, přesouvání úkonů nahoru či dolů
Následné podmínky: Skript je uložen

Tabulka č. 4.: Detail případu užití – editace skriptu

V tomto detailu případu užití z tabulky č. 4 můžeme pohlédnout na to, jak systém pracuje se samotným vytvářením skriptu. Jako vstupní podmínka je zde, že máme vybranou fotografii a je v editačním režimu. Poté musíme spustit samotné nahrávání skriptu. U nahrávání si kdykoliv během editace můžeme vybrat, zda se nám mají veškeré kroky přenášet do skriptu nebo ne jednoduchým výběrem play/stop skript. Pokud se jedná o první spuštění nahrávání a tím pádem vytvoření samotného skriptu, tak se nás aplikace dotáže na její název. Po vytvoření názvu se nám zaznamenávají veškeré námi chtěné změny i do skriptu. Všechny tyto změny lze vidět v pravé dolní části aplikace.
Všechny jednotlivé kroky lze mezi sebou přehazovat posunem nahoru, také je lze z posloupnosti vymazat nebo je editovat. Pro editaci stačí dvakrát provést dvojklik na hodnotu vlastnosti a zobrazí se nám okno, kde vytvoříme novou hodnotu a tuto uložíme.

Ukládání skriptu je navrhnuto tak, aby se celý uložil, pouze pokud si to uživatel vyžádá. Jedná se o odlehčení aplikace na hardwarovou náročnost. Pokud bychom veškeré kroky chtěli okamžitě zaznamenávat, museli bychom neustále v paměti počítače udržovat otevřený dokument. Pokud uživatel chce, může kdykoliv otevřít již dříve uložený skript a tento zpracovávat a upravovat.

4.2.5 Detail případu užití pro tvorbu skriptu

Případ užití: Úpravy skriptu
ID:S2
Účastníci: Uživatel
Vstupní podmínky: Je vytvořen nebo otevřen skript
Tok událostí: 1. systém zobrazí v TreeView strukturu skriptu 2. systém umožní posun nahoru a dolů jednotlivých funkcí ve skriptu 3. uživatel vybere funkci 4. systém posune funkci v pořadí provádění 5. systém umožní smazání jednotlivých funkcí 6. uživatel vybere funkci 7. systém smaže vybranou funkci
Následné podmínky: Skript je editován

Tabulka č. 5.: Detail případu užití – úpravy skriptu

Tento detail případu užití z tabulky č. 5 nám popíše postupné kroky při editaci skriptu. Jako vstupní podmínka je zde nutností jakýkoliv právě vytvořený skript či otevřený kterýkoliv z dříve vytvořených skriptu. Celý proces se odehrává v pravé dolní části editační strany.
Komponenta TreeView nám zobrazuje vzhled skriptu ve stromové struktuře. Systém umožní tři základní editační funkce, samozřejmě vkládání vyjímaje. Tyto možnosti jsou posun funkce o jedno pořadí nahoru, posun funkce o jedno pořadí dolů a smazání funkce. Proto, abychom mohli tyto vlastnosti využít musí uživatel zvolit v TreeView kořenový člen funkce dříve zaznamenané a poté

zvolit některou z editačních funkcí. Při mazání se funkce smaže, při posunu se funkce posouvá před předešlý/následující kořenový člen funkce.

4.2.6 Detail případu užití pro porovnávání

Případ užití: Porovnávání fotografií
ID:S12
Účastníci: Uživatel
Vstupní podmínky: Je zobrazena hlavní strana
Tok událostí: 1. Systém zobrazí editační prvky pro výběr fotografie k porovnávání 2. Systém zobrazí editační prvky pro výběr adresáře k porovnávání 3. Systém umožní nastavení tolerance v pixelech 4. Uživatel spustí porovnávání 5. Systém vytvoří histogram z vybraného obrázku 6. Systém prochází fotografie z vybraného adresáře, vytváří pro ně histogram a porovnává z prvním histogramem 7. Systém vypíše všechny vyhovující stejné fotografie
Následné podmínky: Skript je uložen

Tabulka č. 6.: Detail případu užití – porovnávání fotografií

Tento detail případu užití, který máme v tabulce č. 6, nám popíše kroky pro porovnávání fotografií. Vstupní podmínkou je, že musíme mít spuštěný program a tím mít zobrazenou hlavní stranu programu. Samotné porovnávání se spustí pod nabídkou soubor → porovnávání. Po vybrání této funkce se nám zobrazí okno, kde vybereme fotografii, kterou chceme porovnávat a složku, nad kterou chceme hledat stejné, či silně podobné (možnost nastavení odchylky pixelů) fotografie. Jako poslední položku můžeme nastavit právě odchylku v pixelech na jednotlivé složky histogramu. Po spuštění se na hlavní fotografii vytvoří její histogram. Tvorba histogramu probíhá tak, že se nejprve fotografie převede do stupně šedě a poté se vypočítá histogram. Po vytvoření histogramu začneme procházet vybraný adresář, pro nalezené fotografie vytvoříme taktéž histogram a porovnáme s hlavním histogramem. Pokud nalezneme shodu, název souboru se nám uloží. Po dokončení prohledávání se nám vypíší stejné fotografie.

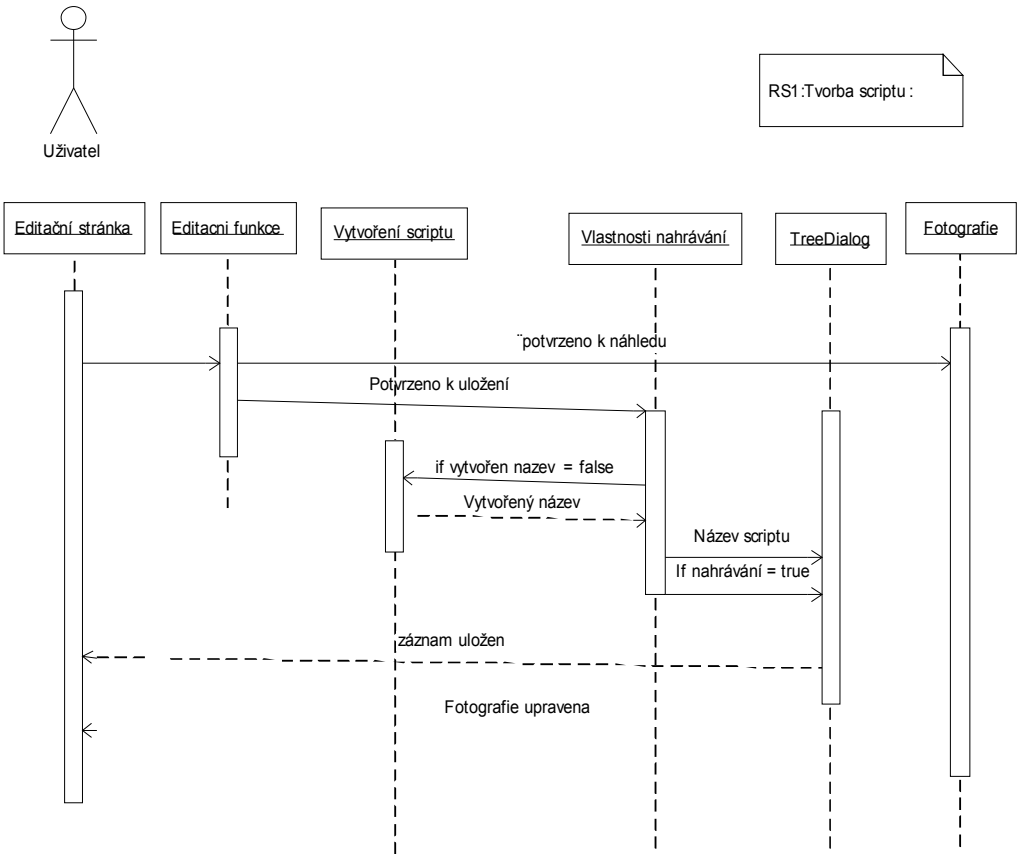
5. Analýza

Jedná se vlastně o rozebrání jednotlivých částí systému, kdy, jak a za jakých podmínek se bude jaká část chovat. Analýza se tvoří z důvodu toho, aby si uživatel ještě předtím, než uvidí samotný funkční systém, dokázal představit, jak se bude chovat. V analýze se tvoří například sekvenční nebo stavové diagramy.

5.1 Sekvenční diagramy

Interakce mezi objekty softwarového systému popisují v jazyce UML sekvenční diagramy. Tyto diagramy postihují, jaké zprávy (požadavky) jsou mezi objekty zasílány z pohledu času. Diagram je tvořen objekty uspořádanými do sloupců a šipky mezi nimi odpovídají vzájemně si zasílaným zprávám. Zprávy mohou být synchronní nebo asynchronní. V případě synchronních zpráv odesílatel čeká na odpověď (odezvu) adresáta, v případě asynchronní zprávy odesílatel nečeká na odpověď a pokračuje ve vykonávání své činnosti. Souvislé provádění nějaké činnosti (operace) se v sekvenčním diagramu vyjadřuje svisle orientovaným obdélníkem. Odezvu adresáta lze opět modelovat, v tomto případe tzv. návratovou zprávou (přerušovaná čára). Tok času probíhá ve směru shora dolů [2]. Všechny sekvenční diagramy naleznete v příloze na CD

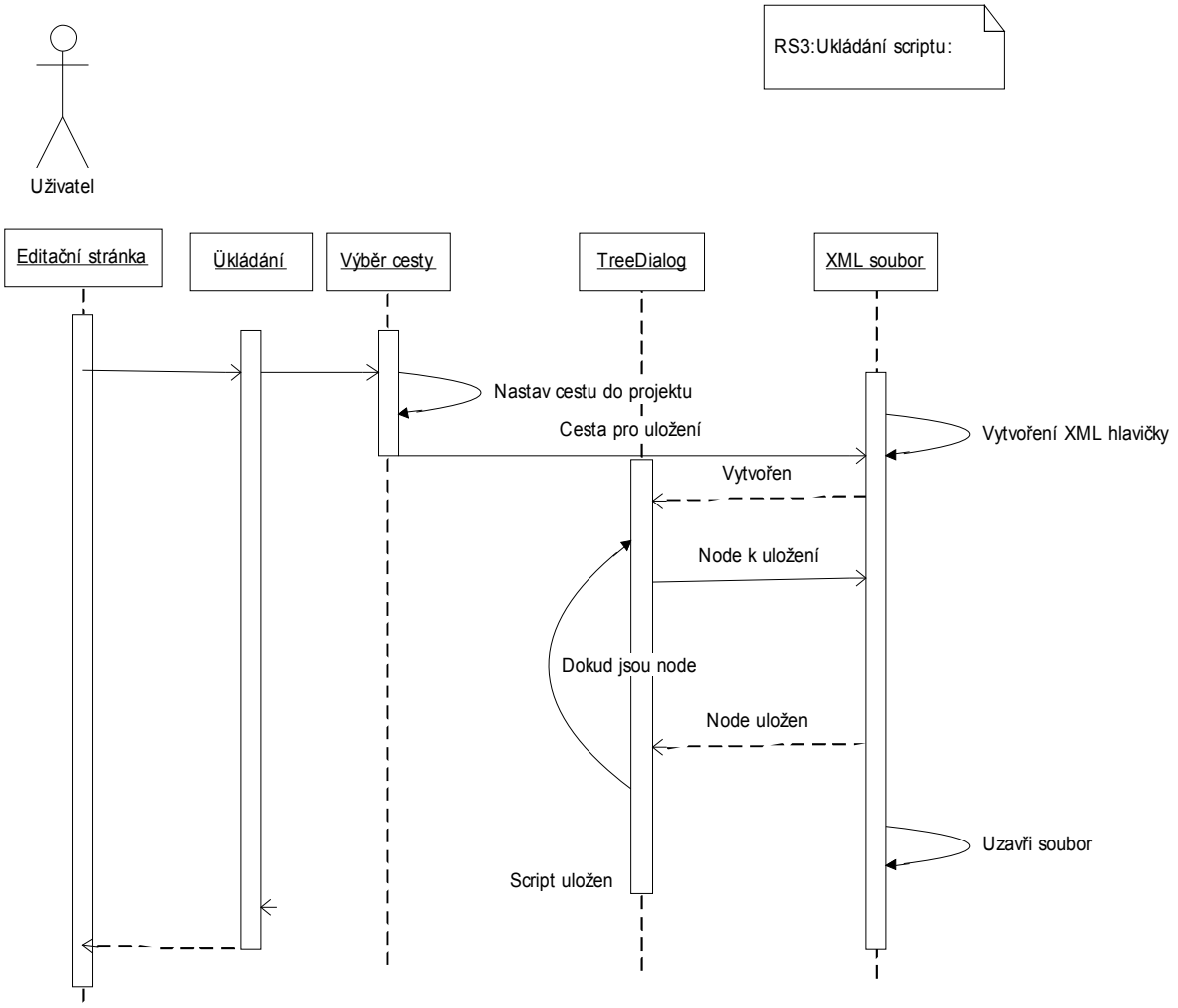
5.1.1 Sekvenční diagram pro tvorbu skriptu



Obrázek č. 2. :Sekvenční diagram – tvorba skriptu

V tomto případě celý sekvenční diagram z obrázku č. 2 zahajuje uživatel. Ostatně veškerou komunikaci a spouštění procesu obstarává z reálného světla jakýkoliv uživatel, protože aplikace je volně přístupná kterémukoliv uživateli. Tento uživatel, přes editační stranu aplikace, která bude celou dobu při vytváření skriptu aktivní, musí v první řadě zahájit spuštění nahrávání sekvence a tím vytvořit název pro skript. Po spuštění nahrávání se zobrazí objekt, do kterého po nás aplikace vyžaduje zapsání názvu. Po jeho potvrzení objekt předá název komponentě TreeDialog, kde se nám budou po celou dobu nahrávání zaznamenávat a zobrazovat veškeré změny. Poté uživatel vybírá samotné funkce, které se mají zaznamenávat a po potvrzení se opět veškeré změny zobrazují do komponenty, současně s potvrzením se změny odešlou na objekt fotky, která se změní dle výběru a změny se zobrazí na editační straně.

5.1.2 Sekvenční diagram pro ukládání skriptu

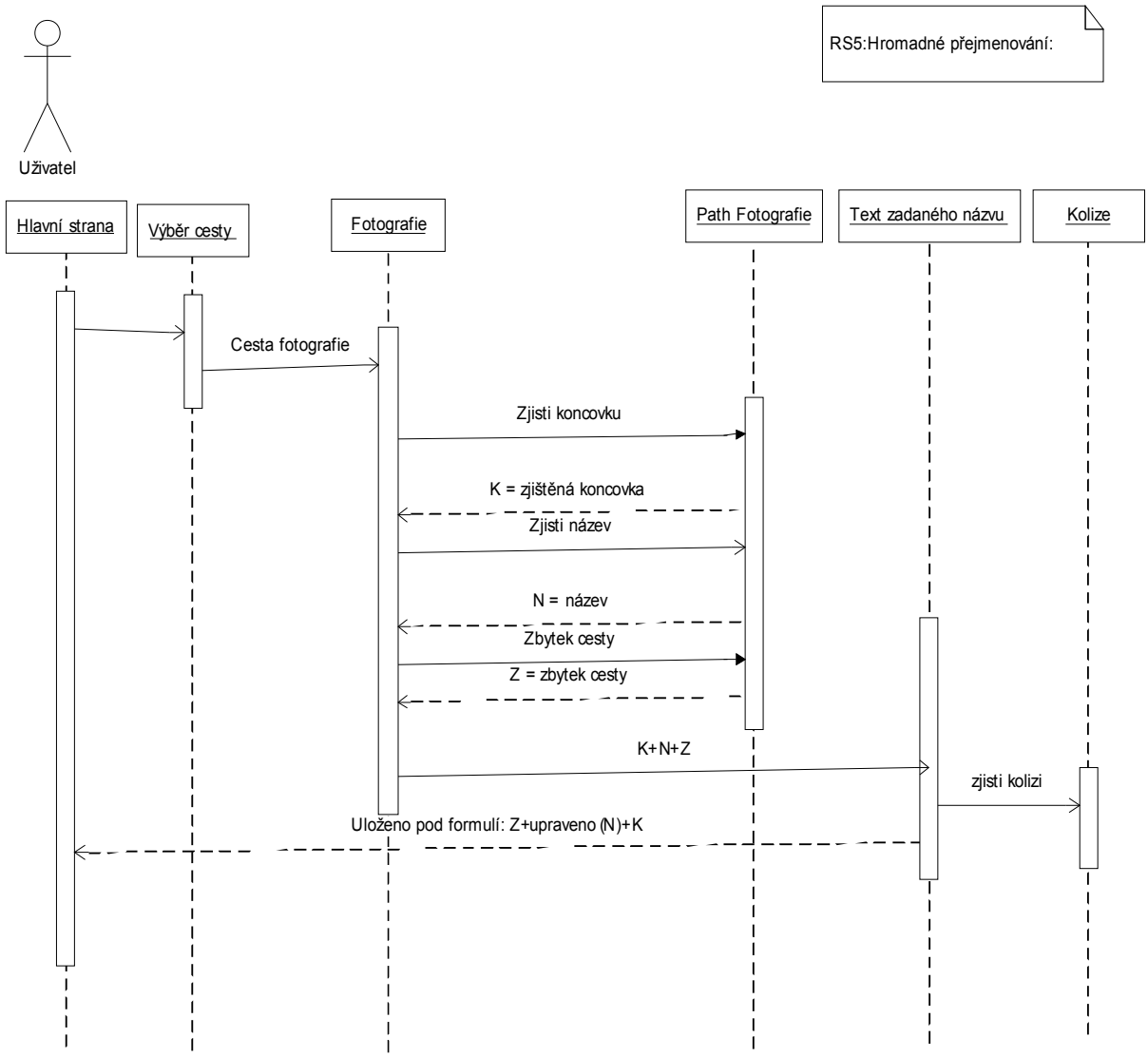


Obrázek č. 3. :Sekvenční diagram – ukládání

V následujícím sekvenčním diagramu z obrázku č. 3 si vysvětlíme, jak se objektově vytváří skript. Vše probíhá na pozadí editační strany, kde tento objekt má živnost po celou dobu vytváření skriptu.

Vstupní podmínkou je, že je vytvořena nějaká sekvence příkazů pro uložení. Uživatel zvolí, že chce provést uložení. Systém zobrazí objekt pro zadání cesty pro uložení, přitom si jako defaultní cestu nastaví do adresáře s projektem. Poté se vytvoří XML soubor, do kterého změny budeme provádět. Do XML souboru se zapíše hlavička pro XML soubory. Následně se začne přenášet obsah TreeDialogu do XML souboru v cyklu, dokud se čtení z komponenty nedostane na konec. Nakonec se XML soubor uzavře, odešle informaci o konci ukládání přes objekt ukládání až k editační straně. Ukládání skriptu je dokončeno.

5.1.3 Sekvenční diagram pro hromadné přejmenovávání



Obrázek č. 4. :Sekvenční diagram – hromadné přejmenovávání

Tento sekvenční diagram na obrázku č. 4 ukazuje, jak pracuje systém, pokud chceme hromadně přejmenovávat fotografie. Přejmenovávání se aplikuje v hromadné úpravě. Tato se spouští přes hlavní nabídku. Pro přejmenovávání první musíme určit, jakou fotografii chceme přejmenovávat a to tím, že načteme její kompletní zdrojovou cestu.

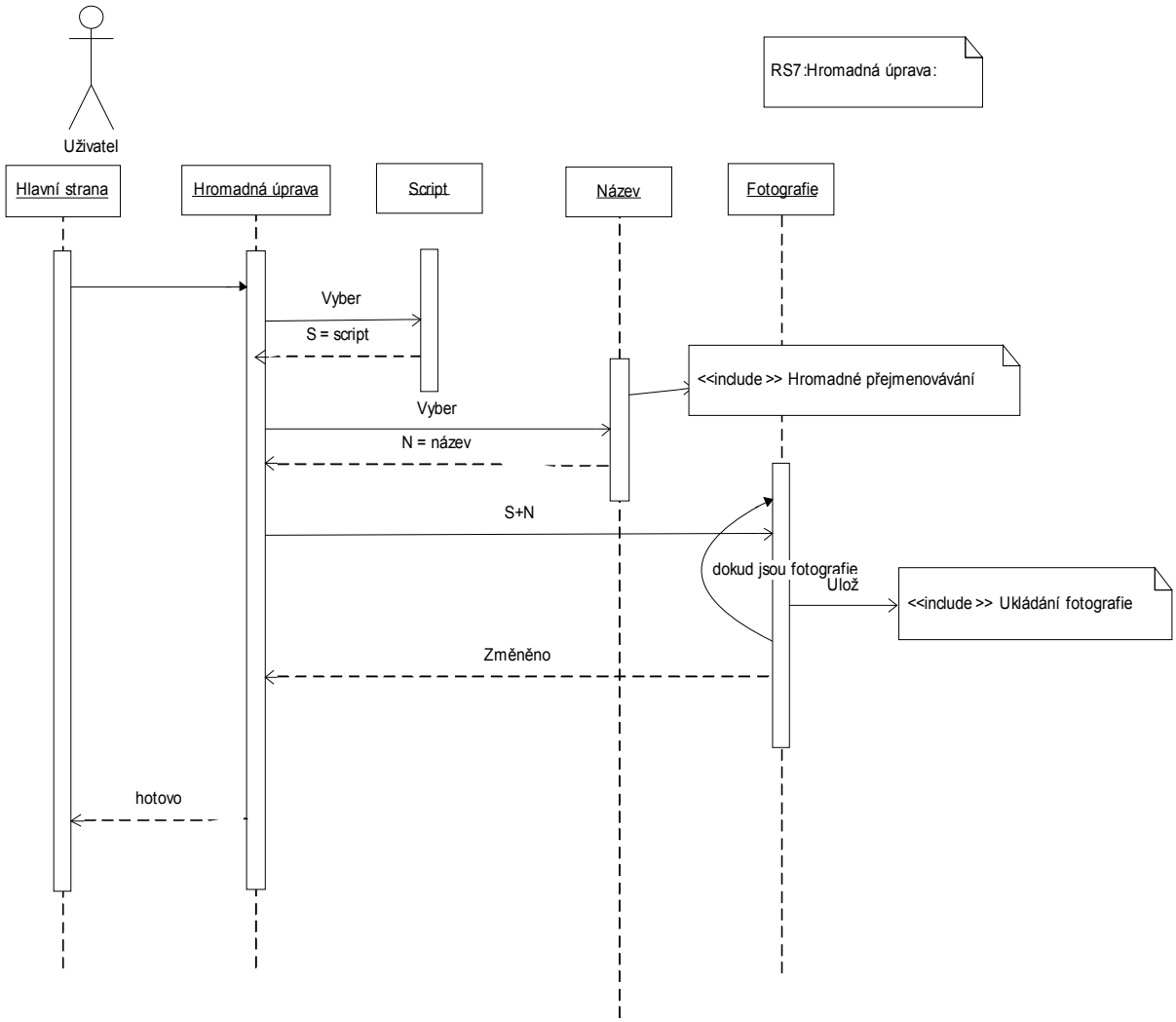
Poté musíme tuto cestu rozebrat tím způsobem, že jí rozdělíme na tři hlavní části a to cesta k fotografii, název fotografie a typ fotografie (její koncovku). Příklady nalezneme v tabulce č. 7.

Kompletní cesta	Cesta k fotografii	název	typ
C:\Documents and Settings\image\foto.jpg	C:\Documents and Settings\image\	foto	.jpg
D:\foto\fotky\P5210128.JPG	D:\foto\fotky\	P5210128	.JPG
D:\foto\Maturita\image.bmp	D:\foto\Maturita\	image	.bmp

Tabulka č. 7.: Příklad rozebrání cesty pro hromadné přejmenovávání

Tyto tři parametry se předají objektu pro vytváření nového názvu. Zde uživatel zadá nový název, přitom může využít původní název. Následně uživatel může předat název pro kontrolu kolize názvů na HDD. Tato kolize kontroluje stejnost názvu a umístění více jak jedné fotografie. Následně se fotografie uloží pod již novým názvem do nového umístění. Název fotografie je změněn.

5.1.4 Sekvenční diagram pro hromadnou úpravu



Obrázek č. 5. :Sekvenční diagram – hromadná úprava

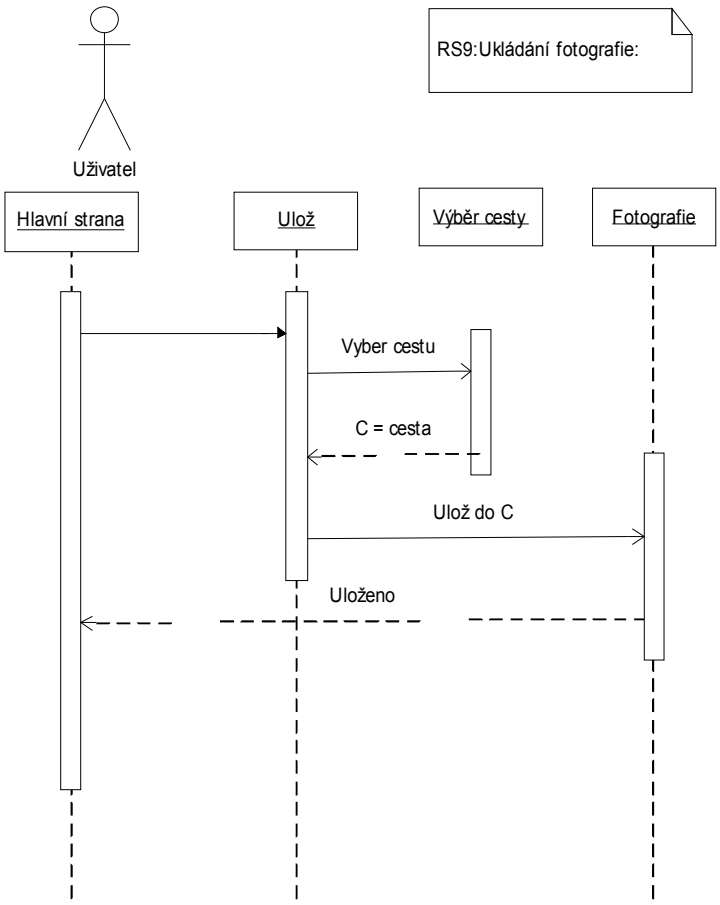
Tento sekvenční diagram z obrázku č. 5 nám poskytne pochopení pro procesovou posloupnost při hromadné úpravě. Celý proces lze spouštět kdykoliv za běhu aplikace, který je schovaný pod objektem hlavní stránky a spouští se zde v hlavní nabídce start → hromadná úprava. Poté se nám zobrazí objekt pro hromadnou úpravu samotnou. Zde budeme využívat hlavních tří objektů.

Ten první se nás dotáže na požadovaný skript, který chceme při úpravě aplikovat. Tato volba je dobrovolná a je čistě na uživateli, zda si přeje některý z dříve vytvořených skriptů využít či nikoliv.

Druhý objekt je pro stanovení názvů obrázků. Tento objekt je samostatně vytvořen a náš sekvenční objekt pouze rozšiřuje <<include>>.

Třetí nejhlavnější objekt je pro dotázání cesty k adresáři, kde se nacházejí fotografie, které chceme upravovat. Fotografie se postupně vybírají a na každou z nich se aplikují první dva objekty dříve vybrané. Jakmile cyklus skončí, hromadná úprava je hotova.

5.1.5 Sekvenční diagram pro ukládání fotografie

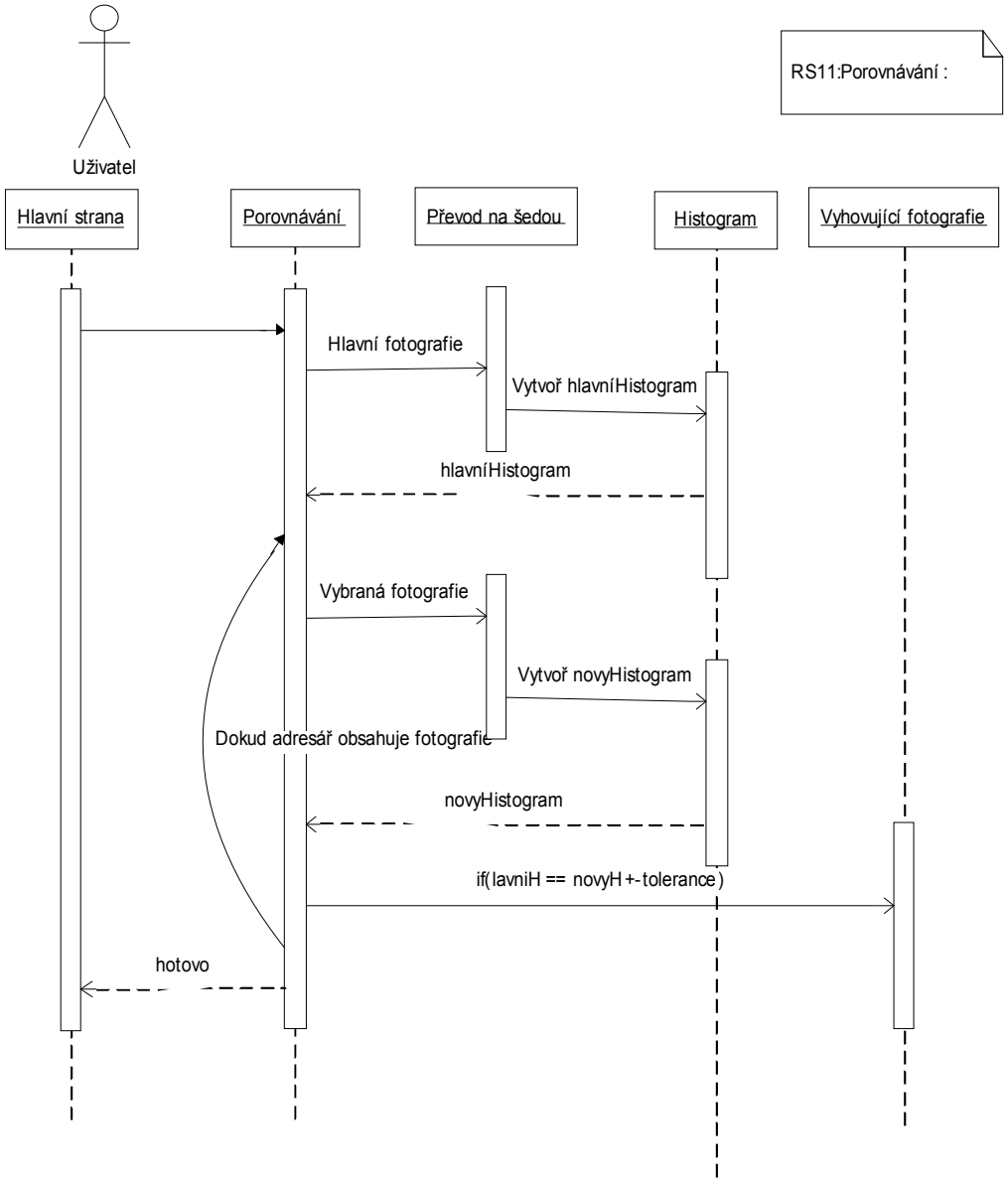


Obrázek č. 6. :Sekvenční diagram – ukládání obrázku

Tento sekvenční diagram na obrázku č. 6 je nejtriviálnější ze všech. Jedná se o klasické ukládání obrázku. Tato operace pracuje na pozadí hlavní strany. Proces se dá spustit kdykoliv, pokud máme vybranou fotografii. Proces se také spouští při hromadné úpravě na každou změněnou fotografii.

Proces je stanoven na dvou základních krocích. V prvním se vybere cesta, do které se má fotografie uložit a ve druhém je fotografie na zadané místo uložena. Výsledkem je uložená fotografie.

5.1.6 Sekvenční diagram pro porovnávání fotografií



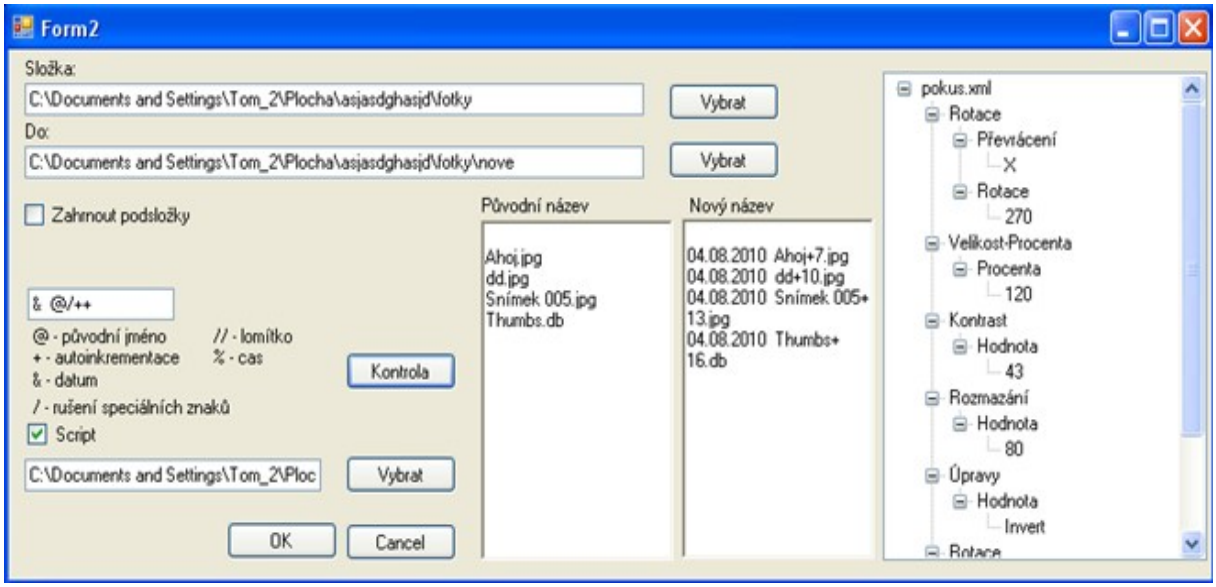
Obrázek č. 7. :Sekvenční diagram – porovnávání fotografií

Sekvenční diagram z obrázku č. 7 popisuje, jak program porovnává fotografie. Proces lze spustit v kterékoliv fázi práce s programem. Po výběru všech potřebných parametrů předává program postupně všechny fotografie k tvorbě histogramu a porovnává je s hlavním histogramem. Pokud nalezneme shodu, tak se zapíše název souboru a na konci se vše vypíše uživateli na obrazovku.

6. Detailní kódové řešení

V této kapitole si detailně popíšeme složitější kódové posloupnosti a netriviální funkce pro následující řešitele. Nebudeme se zde věnovat triviálním funkcím jako otáčení, zvětšování fotografií, či funkcí jasu, kontrastu, rozmazávání a šum. Tyto triviální funkce jsem převzal z internetových zdrojů, protože jsou pro veškeré aplikace totožné a proto je zbytečné je zde řešit. Funkce, kterými se zde budeme zabývat, jsou hlavně všechny součásti hromadné úpravy a tvorba skriptů.

6.1 Hromadná úprava



Obrázek č. 8. :Hromadná úprava

Hromadná úprava nám slouží k aplikování skriptu či přejmenovávání fotografií ze zvoleného adresáře do cílového. Volba pro hromadnou úpravu může vypadat například jako na obrázku č. 8. Zde vidíme kompletně vyplněnou úpravu. Tady se snažíme všechny obrázky z adresáře C:\Documents and Settings\Tom_2\Plocha\asjasdghasjd\fotky upravit tím způsobem, že každou fotku přejmenujeme formulí „& @/++“ která slovně znamená „datum, mezera, původní jméno, plus, inkrementace čísla od jedničky“. Názvy všech obrázků jsme si zkontrolovali a nedošlo k červenému zvýraznění žádného z názvů, tudíž nedošlo ke kolizi. Také zde máme nastavený skript na pokus.xml, jehož kroky vidíme v pravé části obrázku č. 8. Systém postupně vybírá všechny fotky a upravuje je postupně následujícím způsobem.

1. Systém otevře jednu fotku z adresáře
2. Systém pošle fotku se skriptem do editačního režimu kde se fotka upraví dle zadaného skriptu
3. Systém již upravenou fotku přejmenuje dle zadané formule
4. Systém uloží upravenou a přejmenovanou fotku do zadaného cílového adresáře.

6.1.1 Změna názvu

Změna názvu se zahajuje při ukládání obrázku. Kdy se volá:

```
img.Save(cestaBezSouboru + '\\' + ZjistiNazev(file));
```

Název se vlastně skládá ze dvou úkonů. Kde cestaBezSouboru je námi zadaná sílová cesta pro uložení, která je v případě volby průchodu podadresáře rozšířena právě o název onoho podadresáře. Tuhle cestu poté rozšíříme o upravovaný název, toto nám zajistí funkce „ZjistiNazev“.Tato funkce nám v první řadě vezme pouze samotný název a upraví nám ho na text, který je zadaný ve formuli zadané na hlavní straně. Funkci můžeme vidět pod tímto odstavce. Jak lze ve funkci vidět, tak na začátku si vezmeme formuli pro přejmenování a kontrolujeme postupně její znaky.

Kontrola probíhá ve více mnohé podmínce switch. Pokud se některý ze znaků transformace nalezne, tak do finálního názvu vložíme text schovaný za znakem. Pokud se jedná o nějaký jiný nespecifikovaný znak, tak jej vložíme normálně. Jedinou vy jímkou je znak „/“. Je to znak, který ruší speciální symboly. Poté co převedeme název spustíme funkci, která nám zjistí koncovku souboru, kterou je nutné za název souboru přidat. Všechny speciální znaky nalezneme v tabulce č. 8. Příklady transformace názvu nalezneme v tabulce č. 9

Speciální znaky:

Znak	Význam
@	Původní název
+	Auto inkrementace
%	Aktuální čas
&	Aktuální datum
/	Znak pro rušení významů znaků
//	Lomítko - /

Tabulka č. 8.: Hromadné přejmenovávání – speciální znaky

Jako vstupní názvy jsou : PC2434, obrazek1, prvni, druhy, fotka342, fotka111

formule	Výsledek pro PC2434, obrazek1, prvni, druhy, fotka342, fotka111
obrazek c.+	obrazek c.1, obrazek c.2, obrazek c.3, obrazek c.4, obrazek c.5, obrazek c.6
@/+%	PC2434+13:23, obrazek1+13:23, prvni+13:23, druhy+13:23, fotka342+13:23, fotka111+13:23
//&///++	/1.1.2010/+1, /1.1.2010/+2, /1.1.2010/+3, /1.1.2010/+4,/ 1.1.2010/+5, /1.1.2010/+6
ahoj	ahoj, ahoj, ahoj, ahoj, ahoj, ahoj
//+///@	/1/@, /2/@ ,/3/@ ,/4/@ ,/5/@ ,/6/@
/@@/++	@PC2434+1, @obrazek1+2, @prvni+3, @druhy+4, @fotka342+5, @fotka111+6

Tabulka č. 9.: Hromadná přejmenovávání – příklady transformací názvů

```
private string ZjistiNazev(string file)
        {
            string finalniJmeno = null;
            char[] kodovani = textBox3.Text.ToCharArray();
            int delka = kodovani.Length;
            int pomocna = 0;
            while (pomocna != delka)
            {
                switch (kodovani[pomocna])
                {
                    case '/':
                        if (pomocna + 1 != delka)
                        {
                            if (kodovani[pomocna + 1] == '/')
                            {
                                finalniJmeno += '/';
                            }
                            else
                            {
                                finalniJmeno += kodovani[pomocna + 1];
                            }
                            pomocna++;
                        }
                        break;
                    case '@':
                        finalniJmeno += JmenoSouboru(file);
                        break;
                    case '+':
                        finalniJmeno += vypisovecislo;
                        vypisovecislo++;
                        break;
                    case '&':
                        finalniJmeno +=
DateTime.Now.ToString("dd.mm.yyyy");
                        break;
                    case '%':
                        finalniJmeno += DateTime.Now.ToString("hh:mm");
                        break;
                    default:
                        finalniJmeno += kodovani[pomocna];
                        break;
                }
                pomocna++;

            }
```

Dále jak již jsem naznačil v předchozím odstavci musíme zjistit koncovku souboru. Tato funkce přijímá jako vstupní parametr kompletní cestu původního souboru, který měníme. Tuto si převedeme na pole znaků a v cyklu jej procházíme do té doby, dokud nenalezneme tečku, která indikuje začátek koncovky. V případě pokud se jedná o chybu a soubor nemá koncovku, tak se cyklus zastaví ve chvíli, kdy počet opakování překročí délku řetězce. Tu si systém zaznamenává tím

způsobem, že do proměnné „delka“ si na začátku indikujeme délku samotného řetězce a poté se od této proměnné za každý krok v cyklu dekrementuje jednička a tím se nám určuje délka textu bez koncovky. Poté co cyklus skončí, následuje opačný proces. Vezmeme si celý název a délku, která neznačí koncovku a od této délky zaznamenáváme veškeré znaky kompletního souboru do proměnné „koncovka“ až do konce kompletní cesty souboru. Tím máme zjištěnou koncovku.

```
private string ZjistiKoncovku(string file)
        {
            string koncovka = null;

            char[] nazev = file.ToCharArray();
            int delka = nazev.Length - 1;


            while ((nazev[delka] != '.') || (delka < 0))
            {
                koncovka += nazev[delka];
                delka--;
            }
            if (delka == 0)
            {
                koncovka = null;
            }
            if (delka > 0)
            {
                if (nazev[delka] != '.') koncovka = null;
            }
            nazev = koncovka.ToCharArray();
            delka = koncovka.Length - 1;
            koncovka = null;
            koncovka += '.';
            while (delka > -1)
            {
                koncovka += nazev[delka];
                delka--;
            }
            return koncovka;
        }
```

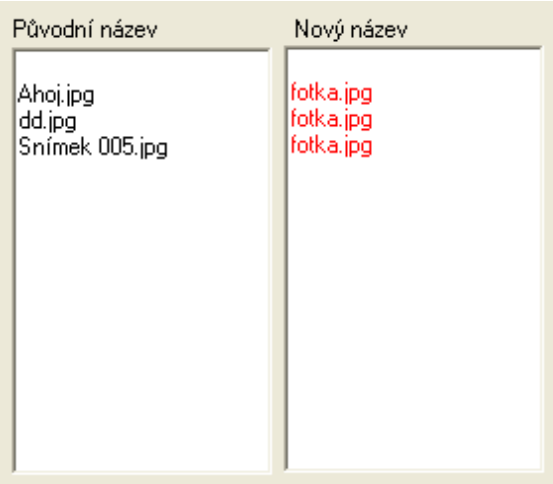
6.1.2 Možný problém hromadného přejmenovávání

Pokud spouštíme hromadné přejmenovávání na složku, která obsahuje více jak jeden soubor, tak mohou nastat problémy. Problém nastane pokud splníme původní zmíněnou vlastnost a pokud do formule pro přejmenování nenastavíme inkrementaci či původní název fotografie. Pro předčasné odhalení tohoto problému zde máme funkci kontroly původního a budoucích názvů. Spuštění kontroly se provádí na formuláři pro hromadnou úpravu. Toto můžeme vidět na obrázku č. 8. Pokud by v názvech byla kolize, tak systém kolizní názvy označí červeným písmem.
Toto můžeme vidět na obrázku č. 9. Uživatel zadal formule pro změnu názvu jako „fotka“ a jelikož zde nemáme ani inkrementaci, ani využití původního názvu, tak nám systém změněné názvy označí červeně.

Kódově je tohle řešeno tak, že se veškeré názvy, které jsou stejné, zaznamenávají do ArrayListu a poté všechny hodnoty, které jsou uloženy v tomto listu, jsou v cyklu procházeny a změněny na červenou barvu.

```
private void ZmenaBarvy()
        {
            foreach (int a in al)
            {
                int lineNumberToSelect = a + 1;
                int start =
richTextBox1.GetFirstCharIndexFromLine(lineNumberToSelect);
                int length = richTextBox1.Lines[lineNumberToSelect].Length;
                richTextBox1.Select(start, length);
                richTextBox1.SelectionColor = Color.Red;
            }

        }
```



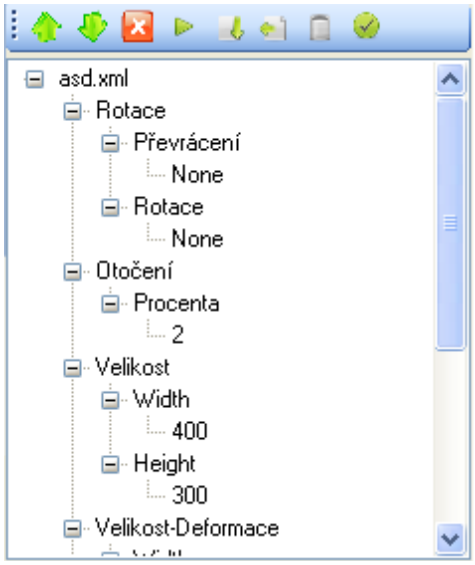
Obrázek č. 9. :Kolize změny názvu

6.2 Vytváření skriptů

Při vytváření skriptů jsem se rozhodoval mezi dvěma variantami možného vytváření. První z nich bylo, že by se automaticky při najetí s fotografií do editačního režimu vytvořil XML soubor a veškeré změny by se prováděly nad tímto souborem přes některou z komponent, která by měla pouze zobrazovací smysl. Tuto variantu jsem ale nevolil vzhledem k tomu, že by se v paměti neustále musel udržovat otevřený dokument, nebo ho neustále otvírat a zavírat.

Proto jsem se rozhodl pro druhou a z mého pohledu sice implementačně složitější, ale pro běh aplikace mnohem vhodnějšího způsobu vytváření. Spočívá v tom, že se zde po celou dobu vytváření přistupuje ke komponentě TreeView.

Tato komponenta nám zobrazuje veškeré změny sama na sobě a veškeré editační funkce pro skripty také pracují na této komponentě. Poté co vytvoří posloupnost kroků je na uživateli, zda si skript uloží. Toto je další výhodou oproti prvnímu způsobu. Nebudeme mít zbytečně uložené skripty, ale pouze ty, které si přejeme. Prostředí pro tvorbu skriptu můžeme vidět na obrázku č. 10.



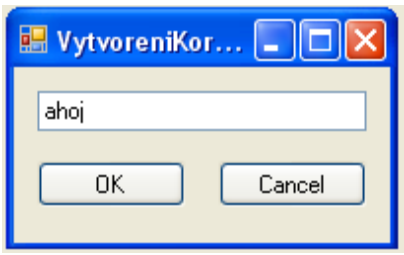
Obrázek č. 10. : Prostředí pro tvorbu a editaci skriptu

6.2.1 První spuštění

Nahrávání skriptu se dá kdykoliv během jeho vytváření opětovně zastavovat či spouštět. Pokud je nahrávání spuštěné, tak nám komponenta TreeView bude zaznamenávat změny v editaci fotografie. Pokud je nahrávání pozastaveno, tak se změny v editaci budou projevovat pouze na samotné fotografii. Jestliže se tedy uživatel rozhodne vytvářet skript a spustí nahrávání, tak se nás systém dotáže na jeho jméno. Toto okno můžeme vidět na obrázku č. 11.

Poté se nám v komponentě TreeView zobrazí zadané jméno, pod kterým se nám budou ukládat všechny další kroky. I o přenesení názvu skriptu do komponenty se nám stará okno z obrázku č. 11, kdy si vytvoří instanci na editační okno a poté nahraje všechny změny přes tuto instanci:

```
      Form1 mp = (Form1)Application.OpenForms["Form1"];
             mp.hlavniKoren =  mp.treeView1.Nodes.Add(textBox1.Text +
".xml");
             mp.vytvoreniScriptu = true;
             mp.ZaznamenavaniScriptu();
             this.Close();
```



Obrázek č. 11. : Zadávání názvu skriptu

6.2.2 Průběh zaznamenávání

Pokud máme vytvořen skript a je spuštěno nahrávání, tak se nám mají veškeré požadované a potvrzené změny zaznamenávat postupně do sekvence kroků pro skript. Po každé vybrané funkci pro úpravu obrázku toto musíme potvrdit tlačítkem, které nám v prvním kroku zobrazí změnu na obrázku a poté se zavolá formulace pro uložení kroku do posloupností kroků pro skript. Příklad tohoto volání vypadá takto:

```
if (nahravaniScriptu) PridaniScriptu2("Jas", "hodnota",
trackBar3.Value.ToString(), null, null);
if (nahravaniScriptu) PridaniScriptu2("Velikost", "Width", textBox3.Text,
"Height", textBox4.Text);
```

Tedy toto volání předává pět parametrů. První z nich je název samotné funkce. Druhá a čtvrtá hodnota jsou identifikátory funkce. Třetí a pátý jsou hodnoty identifikátorů. Funkce je vytvořena univerzálně a je jí možné použít jak pro jeden tak i pro dva parametry funkce.

```
private void PridaniScriptu2(string root, string child1, string child1_1,
string child2, string child2_1 )
        {

            TreeNode prvni = hlavniKoren.Nodes.Add(root);
            TreeNode druhy1 = prvni.Nodes.Add(child1);
            druhy1.Nodes.Add(child1_1);
            if (child2 != null)
            {
                TreeNode druhy2 = prvni.Nodes.Add(child2);
                druhy2.Nodes.Add(child2_1);
            }
        }
```

Po zavolání funkce se nám vytvoří pod hlavní kořen (název skriptu) kořen nový, který označuje název funkce, jako dítě se mu vytvoří název hodnoty a pod něj se vytvoří samotná hodnota. Poté nastává rozhodnutí zda je funkce se dvěma hodnotami či nikoliv. Pokud má funkce pro úpravu obrázku pouze jednu hodnotu, tak funkce PridavaniScriptu2 končí. Jestliže ale funkce pro úpravu obrázku má dvě hodnoty, tak se do kořene názvu funkce vloží druhá hodnota.

6.2.3 Spuštění skriptu

Pokud chceme spustit skript pouze na jednu fotografii, nebo pokud chceme vidět výsledky vzhledů skriptů, aniž by se změny uložily, můžeme toto provést v editačním panelu vytváření skriptu. Toto můžeme vidět na obrázku č. 9.

Poté se začne postupně procházet komponenta TreeView, přečte si jednu větev a tu provede tím způsobem, že se nám vytvoří ukazatel (Ienumerator) na hlavní kořen a postupně se tímto ukazatelem prochází první vrstva podkořenů na hlavní kořen. Znova se vytvoří ukazatel, tentokrát na podkořen. Poté se ve funkci switch vybere, o jakou funkci se jedná a načte do proměnných její hodnoty. Poté se volá funkce na provedení operace.

```
public Image provedScript()
        {
            int operace = 0;
            string prvniHodnota = null;
            string druhaHodnota = null;
            if (vytvoreniScriptu)
            {
                IEnumerator ie = hlavniKoren.Nodes.GetEnumerator();
                 while (ie.MoveNext())
                {
                    TreeNode korenOperace = (TreeNode)ie.Current;
                    IEnumerator ie2 = korenOperace.Nodes.GetEnumerator();
                    ie2.MoveNext();

                    prvniHodnota = null;
                    druhaHodnota = null;

                    switch (korenOperace.Text)
                    {
                        case "Rotace":
                            operace = 1;
                            break;
                        case "Otočení":
                            operace = 2;
                            break;
                       default:
                            MessageBox.Show("Pokazilo se vybrání operace");
                            break;
                    }

                    if (korenOperace.GetNodeCount(true) == 2)
                    {
                        TreeNode a = (TreeNode) ie2.Current;
                        prvniHodnota = a.LastNode.Text;

                    }
                    if (korenOperace.GetNodeCount(true) == 4)
                    {
                        TreeNode a = (TreeNode)ie2.Current;
                        prvniHodnota = a.LastNode.Text;
                        ie2.MoveNext();
                        TreeNode b = (TreeNode)ie2.Current;
                        druhaHodnota = b.LastNode.Text;
                    }
                    ProvedOperaci(operace, prvniHodnota, druhaHodnota);
                }
        }
```

Funkce Proveď operaci je několik funkcí switch, které pouze zjistí jaká operace se má provést, zavolá jí k provedení s hodnotami a případně nastaví veškeré komponenty jako textboxy a checkboxy na hodnoty, které se změnili. Například:

```
case 3: //velikost
                    textBox3.Text = prvniHodnota;
                    textBox4.Text = druhaHodnota;
```

```
                    this.resizeImageXY(int.Parse(textBox3.Text),
int.Parse(textBox4.Text));
                    break;
```

6.2.4 Ukládání skriptů do XML

Poslední funkcí, kterou si zde popíšeme, je přenos skriptu do XML souboru. Funkce se opět pouští v editačním skriptování. Ukládání je nastaveno tak, že se automaticky ukládá do adresáře s aplikací, tudíž se uživatele vůbec neptá na umístění pro uložení. Nejprve se vytvoří StreamWriter, který vytvoří soubor, do kterého se následně uloží XML hlavička. Následně se jako v předchozí podkapitole vytvoří ukazatel na hlavní kořen v TreeView, který prochází ostatní podkořeny, které se ukládají do dokumentu.

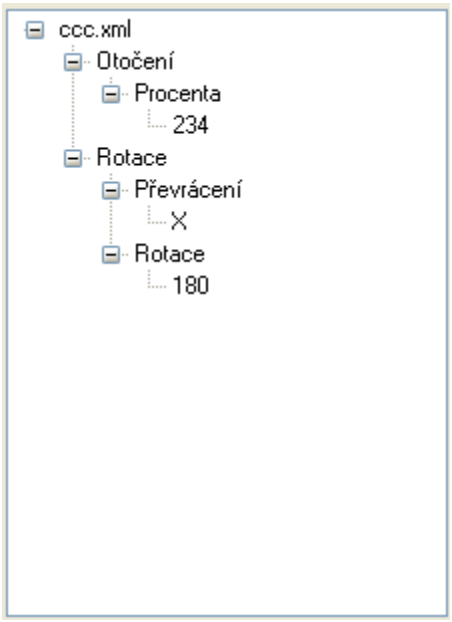
```
 private void exportToXml(string filename)
        {
            sr = new StreamWriter(filename, false,
System.Text.Encoding.UTF8);
            sr.WriteLine("<?xml version=\"1.0\" encoding=\"utf-8\" ?>");

            IEnumerator ie = treeView1.Nodes.GetEnumerator();

            if (ie.MoveNext())
            {
                TreeNode tn = (TreeNode)ie.Current;
                sr.WriteLine("<" + tn.Text + ">");
                parseNode(tn);
            }

            sr.Close();
        }
```

Funkce parseNode opět tvoří ukazatele na pod-kořeny, které prochází a ukládá je jako elementy do XML souboru. Jediné v čem se funkce musí rozhodnout je to, kolik má pod-kořen dětí a podle toho provést počet zápisů do souboru. Například převedení obrázku č. 12 na XML soubor může vypadat takto:



Obrázek č. 12. : Příklad sekvence příkazů

```
<?xml version="1.0" encoding="utf-8" ?>
<ccc.xml>
<Otočení><Procenta>234</Procenta>
</Otočení>
<Rotace><Převrácení>X</Převrácení>
<Rotace>180</Rotace>
</Rotace>
</ccc.xml>
```

7. Závěr

Bakalářská práce byla zaměřena na seznámení se se současnými aplikacemi pro správu fotografií a jejich využívání. Díky těmto poznatkům jsem měl vytvořit vlastní aplikaci, která dokáže jednoduché úpravy s fotografiemi, tvorby skriptů s posloupností kroků úprav nad fotografiemi a hromadné zpracovávání fotografiemi vytvořenými skripty. Poté vytvořit analýzu systému a provést co nejlepší dokumentaci pro následného řešitele. Analýza spolu s dokumentací by měla poskytnout dostatečné informace pro pochopení běhu a funkčnosti aplikace. Všechny vytvořené části jsou uloženy na přiloženém CD spolu s aplikací a kompletní analýzou, jelikož některé triviálnější diagramy jsem zde nerozebíral.

Výsledná aplikace neobsahuje nijak rozsáhlou škálu funkcí pro úpravu, ale je dostatečná pro nenáročné uživatele. Proto směr rozšiřování aplikace bych viděl v rozšíření o nové funkce a možnost dodělání přívětivějšího prohlížeče samotných fotografií bez jejich úprav.

Seznam obrázků

Seznam tabulek

Literatura

[1] VONDRÁK, CSC., Prof. Ing. Ivo. *METODY SPECIFIKACE SOFTWAROVÝCH SYSTÉMU : Funkcní specifikace systému* [online]. Ostrava : VŠB, 2005. 65 s. VŠB - TU Ostrava. Dostupné z WWW: <http://vondrak.cs.vsb.cz/download/>.

[2] VONDRÁK, CSC., Prof. Ing. Ivo. *METODY SPECIFIKACE SOFTWAROVÝCH SYSTÉMU : Interakce a dynamické chování objektu* [online]. Ostrava : VŠB, 2005. 65 s. VŠB - TU Ostrava. Dostupné z WWW: <http://vondrak.cs.vsb.cz/download/>.

[3] Wikipedie : [online]. 1995 [cit. 209-05-05]. ACDSee. Dostupné z WWW: <http://cs.wikipedia.org>.

[4] Wikipedie : [online]. 1995 [cit. 2010-05-02]. Adobe Photoshop. Dostupné z WWW: <http://cs.wikipedia.org>.

[5] KAČMÁŘ , Dalibor. *Programujeme .NET aplikace ve Visual Studiu .NET* . Holandská 8, 639 00 Brno : Computer Press, 2004. 344 s. ISBN 8072265695.

[6] MICHAL, Dobreš. *Zpracování obrazu a algoritmy v C#*. Praha 10 : BEN-Technická literatura, 19-07-2008 . 143 s. ISBN 978-80-7300-2.

[7] NAGEL, Christian, et al. *C# 2008 : Programujeme profesionálně*. Holandská 8, 639 00 Brno : Computer Press, duben 2009. 1904 s. ISBN 978-80-251-2401-7.

[8] *C# Corner* [online]. Image Processing in C#. 2004 [cit. 2004-04-13]. Dostupné z WWW: <http://www.c-sharpcorner.com/>.

[9] Projekty SIPVZ : Programování se zaměřením na .NET a jazyk C# [online]. Programování Windows Forms pomocí C#. 2006 [cit. 2006-10-18]. Dostupné z WWW: <http://projektysipvz.gytool.cz>.

[10] Switch On The Code : [online]. 2007 [cit. 2007-02-14]. C# Tutorial. Dostupné z WWW: <http://www.switchonthecode.com>.

[11] Code Source : [online]. 2008 [cit. 2008-05-02]. Microsoft .NET. Dostupné z WWW: <http://www.codersource.net>.

[12] MEUNIER, Serge. Centre of the Universe : [online]. 2006 [cit. 206-01-22]. Image Processing in C#. Dostupné z WWW: <http://www.sergemeunier.com>.

[13] Digi Neff [online]. 2010 [cit. 2010-04-13]. Editujeme. Dostupné z WWW: <http://www.digineff.cz/>.

[14] Adobe Lightroom, profesionální editor RAW fotografií. Grafika On-Line [online]. 2006, 342, [cit. 206-08-11]. Dostupný z WWW: <http://www.grafika.cz/>.

Obsah CD

dokumenty/PripadUziti.vsd
dokumenty/sekvencnidiagramy.vsd
dokumenty/DetailPripaduUziti.odt
aplikace/